Programming assignment: eye tracking
April 2018

Eye tracking is an important component in VR/AR applications, since it allows a more natural interaction by knowing where user is looking at. This assignment consists of two tasks related to pupil-based eye tracking techniques: 1) find and describe the pupil (approximated as a 2D ellipse) in an eye image; 2) find the 3D location and orientation of pupil (approximated as a 3D circle) given its observation in an eye image.

TASK 1. Pupil detection

The input in this task is an eye image and the output will be the mask image showing the detected pupil. (See Fig.1)

Goal

- Detect the contour of the pupil and fit a 2D ellipse, given an image I that contains an eye image.

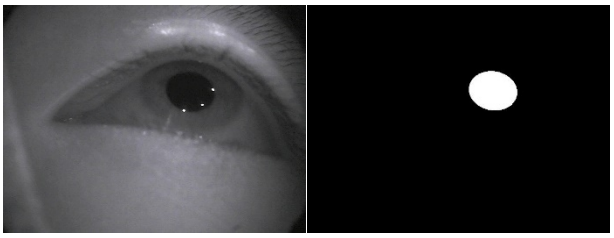

Figure 1 An example of pupil detection. Left: Eye image. Right: Mask of detected pupil

Deliverables

- C++ code, executable and a README file on how to run the program for the pupil detection.

TASK 2. Pupil pose estimation

In this task, we reformulate the problem with a few assumptions and synthetic data. First, the camera is a pinhole camera and the intrinsic matrix K is known. Second, we assume the pupil can be described as a 3D circle $(\boldsymbol{c}, \boldsymbol{n}, r)$, where $\boldsymbol{c}$ is a 3D point corresponding to the circle center, $\boldsymbol{n}$ is the direction orthogonal to the circle, and r is the circle radius. The 3D circle corresponds to the intersection between an unknown arbitrary plane and the surface of a known 3D sphere $(\boldsymbol{C}, R)$, where $\boldsymbol{C}$ is a 3D point corresponding to the sphere center and R is the sphere radius. Please note that the center of the 3D circle $\boldsymbol{c}$ is not on the surface of the 3D sphere in the general case.

The input of the task is a synthetic pupil image, the intrinsic camera matrix, and the parameters of the known 3D sphere $(\boldsymbol{C}, R)$.

Goal

- Use the code from Task 1 to detect the pupil and find its corresponding 2D ellipse.
- Given the camera matrix and 3D sphere, draw the boundary of the projected 3D sphere on the image plane.
- Find the parameters of the 3D circle $(\boldsymbol{c}, \boldsymbol{n}, r)$ describing the pupil.

Deliverables

- C++ code, executable and a README file on how to run the program for the pupil pose estimation.

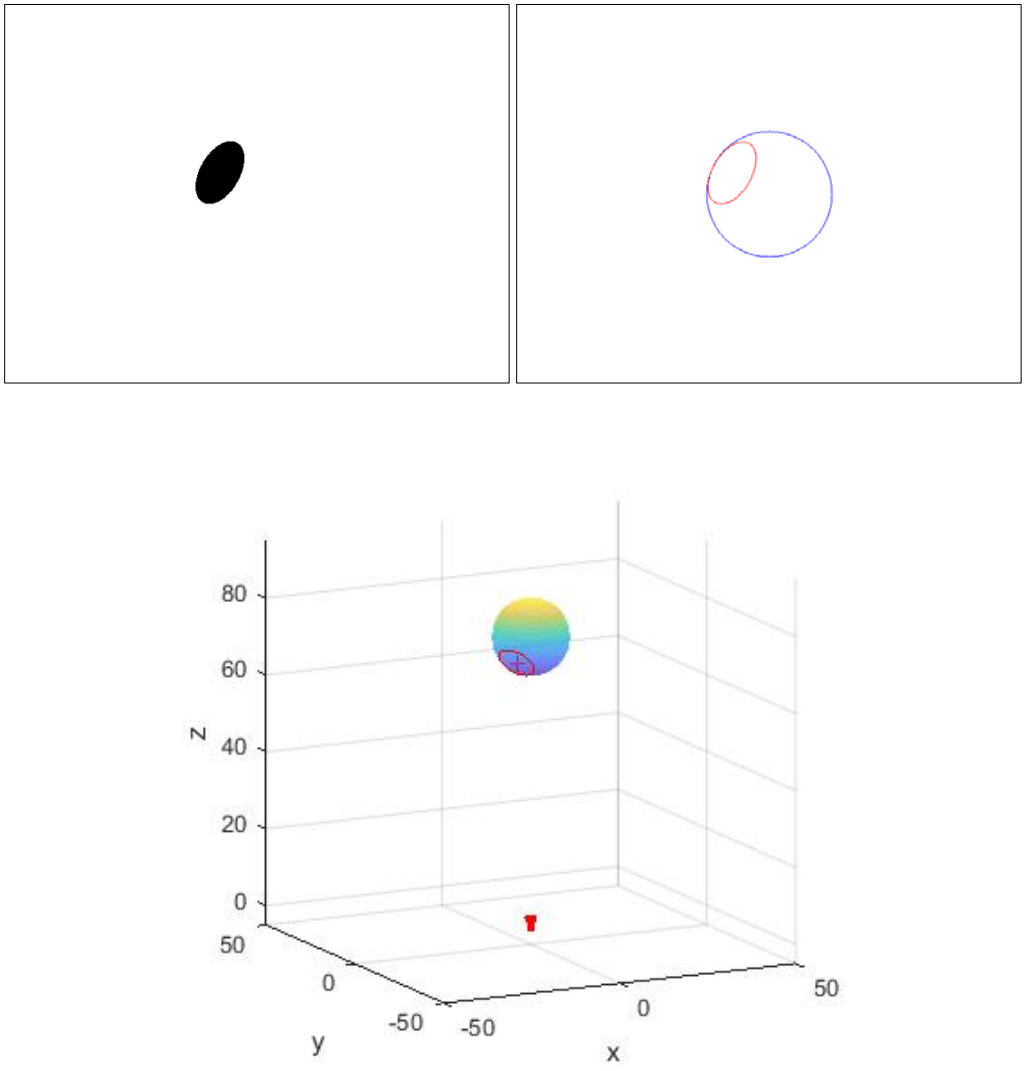

Figure 2 Top Left: Synthetic pupil image. Top Right: Pupil (red) and projected 3D sphere (blue). Bottom: An illustration of the locations of the known 3D sphere and pupil (red 3D circle) w.r.t. camera.